UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/700,224	11/03/2003	John H. Sandham	GB920095006US1	3106

48916 7590 09/28/2009

Greg Goshorn, P.C.
9600 Escarpment
Suite 745-9
AUSTIN, TX 78749

EXAMINER	
VO, TED T	

ART UNIT	PAPER NUMBER
2191	

MAIL DATE	DELIVERY MODE
09/28/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/700,224	SANDHAM ET AL.	
	Examiner	Art Unit	
	TED T. VO	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *06 July 2009*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,3-7,10,11,13-19 and 100-111* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,3-7,10,11,13-19 and 100-111* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the amendment filed on 07/06/2009.

Claims 2, 8-9, 12, 20-99 are canceled. Claims 100-111 are new.

Claims 1, 3-7, 10-11, 13-19, 100-111 are pending in the application.

Response to Arguments

2. This is in response to the arguments remarks filed on 07/06/2009:

First of all, Examiner recognizes a lot of passages from the Office Action with typo errors, as being indicated with "[sic]" in the applicants' remarks. It would be expected the arguments in the manner of patentability. Regarding the Applicant's statement for clarifying the claimed term "subject code", Applicant pointed out that the code that is emulated is subject code, and the outcome of the emulation is target code. The Applicant's clarification will be provided for the Examiner's interpretation in the action.

In regard to the arguments to the claims rejected under Barrio, it appears one of the arguments is that "[B]arrio is not executing a block of subject code through the emulator", but "[B]arrio describes executing not blocks of code but rather instruction line-by-line", but, "[B]arris describes executing not blocks of code but rather instructions line-by-line. The blocks of code described correspond to the interpreter in which a case statement translates "OPCODE1"

to "opcodelO;" "OPCODE2" to "opcode20:" and so on. In this example, the interpreter is executing in blocks however the interpreted code is executed one instruction at a time".

Applicants also depicted a passage in Barrio and argued Barrio is not comparing the results of execution of blocks of code.

Examiner's response: As mentioned by Barrio in the reference, emulation is "try to be equal or better than someone or something". Based on the definition, it shows that an emulator is such a "Virtual machine". In the reference (p. 20-24), it mentions various types of emulators, such as a CPU emulation core, I/O emulations, Graphic Hardware emulation, etc. Thus, an emulator is a computer program that tries to be equal or better than something. Take an Interpreter as for the CPU emulation, the task of a CPU emulation is to interpret "code" (therefore: Examiner interprets this code is "subject code"; this code clearly shown in Barrio via the definition of Emulator) so that the interpreted code ('target code') can be executable as the nature machine language of a computer. It is obvious that no interpreter is perfect. Therefore, an emulator needs to be tested. The item 8 (in p. 24-25), testing the emulator, which is about the verification of emulations. This has been discussed through out the reference. Especially, in the item 8, it mentions testing the CPU emulator, by taking the real code that is received by the CPU (See in second paragraph in item 8: "[m]ean to generate all the possible instructions which can receive the CPU..") [herein claimed as: ("*native machine state from execution of the one block of subject code natively on the subject processor*"], and comparing to the result in the emulator (See also in second paragraph, ("[c]ompare the result in the emulator with the real result"). Therefore, it should understand that the emulated result and state of the CPU emulator will be compared against a real result and its state of a native code if it is for the real CPU.

Since the claims appear in the area of code translation which is a type of CPU emulation, its subject code is divided into blocks is only for conforming to the act of compilation and the target CPU resources. Thus, it eases the translation. This is mentioned by Barrio in section 2.2, CPU emulator,

> "[B]inary translation means to get the native code for the emulated CPU and translate this code, using techniques related with compilation for example, into code for the target CPU. The translated code is stored and executed every time that is called the emulation of the CPU.
> The translation is performed in blocks of code, sometimes related with the concept of basic block in compiler theory sometimes not. The reasons for translating in blocks rather than translating the full program at once are diverse. From the fact that not all the code can be known at first, to the fact that there must be points for stopping the emulation and to ease the process of translation, which is faster working in a block base."

And since every CPU emulator tries to implement the CPU native code, it must include a task for decoding the emulated code for the CPU instructions. The statement for words in the claim, "executing a block of subject code", clearly is only a generic statement that is for executing line-by-line within a block of code; this is a mere common statement in expression. Since every execution is controlled by a clock, instructions must be executed line-by-line. Therefore, in the remarks, Applicants highlight "**All words in a claim must considered**"; this is only for the **words** in judging the **patentability**. None of such words have been pointed out by applicants; while the numbers of words in the claims exceed the numbers of words from p. 4 through p. 18 in the specification. There are a lot of words in the claims that are not in the specification; thus, the words in the claims admit the conformity rather than address the patentability. Applicants further argued to the rejections of dependent claims 4, 10, 13, 17, and 19; however, the arguments are not persuasive since the limitations recited in the claims do not show novel feature. They are explicitly or implicitly addressed in Barrio as cited.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

4. Claims 1, 3-7, 10-11, 13-19, 100-111 are rejected under 35 U.S.C. 102(b) as being anticipated by Barrio, "Study of the techniques for emulation programming", The Technical University of Catalonia (UPC), Europe, pp. 1-152, 6-2001 (http://personals.ac.upc.edu/vmoya/docs/emuprog.pdf).

As per Claim 1(Currently amended): Barrio discloses,

A method of verifying program code conversion performed by an emulator, comprising:

a) dividing a subject code into a plurality of blocks (p. 19: blocks of code) ***and executing one of the blocks of subject code through an emulator in a process image on a subject processor according to an emulation context up until a comparable point in the subject code to provide an emulated machine state;*** (See p. 18-19. both sec. 2.1 and sec. 2.2: see, See the program in p. 18, an Interpreter emulator, that uses case switch to divide the different opcodes ('subject code'). These verifying program code of opcodes (emulator) are divided by the Interpreter emulator, and each OPCODEi() is executed till the point break (*'a comparable point'*). Further see "translation is performed in blocks of code" in p. 19)

b) performing a context switch to a native context (See p. 43: See GetContext(); e.g. OPCODEi()) ***and executing the same block of subject code natively in the same process image on the same subject processor up until the same comparable point in the subject code to provide a native machine state*** (See the program call OPCODEi(), that call to a native state of the opcode. See sec, 2.1, and 2.2); ***and***

c) comparing the native machine state from execution of the one block of subject code natively on the subject processor against the emulated machine state from execution of the same block of subject code on the same subject processor through the emulator at the comparable point in the subject code;
wherein the native machine state includes a native memory image and the emulated machine state includes an emulated memory image and (a) and/or (b) includes selectively isolating access to a memory associated with the subject processor to obtain the native memory image and/or the emulated memory image, respectively.
See page 18, including the program, and see the beneath paragraph of the program. Also see p. 24, sec. 8: Testing the emulator.

As per Claim 3: This claim is indefinite. Barrio discloses, *The method of claim 1, comprising performing (a) prior to performing (b).* (Incorporated with the rejection of claim 1, Barrio's program shows it OPCODEi() execution is prior to a context switch).

As per Claim 4: Barrio discloses, *The method of claim 3, wherein: (a) further comprises providing an emulated image of the subject processor* (Still referred to the program of p. 18, and sec. 2.1, and 2.2, and further see testing the emulator in p. 24, sec. 8); *(b) further comprises providing a native image of the subject processor following the native execution of the program code* (i.e. real result run on real CPU emulator (p. 24, or p. 62), *following the native execution of the program code; and (c) further comprises comparing the emulated image of the subject processor against the native image of the subject processor* (See p. 24, and p. 62, i.e. 'compared')

As per Claim 5: Barrio discloses, *The method of claim 4, wherein (a) comprises providing the emulated image of the memory in a load/store buffer associated with the memory, such that the memory is not affected by executing the subject code through the emulator* (i.e. the OPCODEi() defined by the program according to the CPU emulation: *emulated image of the memory*, and see p. 22 first paragraph).

As per Claim 6: Barrio discloses, *The method of claim 5, wherein the emulated image of the subject processor includes an image of one or more registers* (i.e. CPU emulation, p. 22).

As per Claim 7: Barrio discloses, *The method of claim 6, wherein the emulated image of the subject processor includes an image of one or more condition code flags* (e.g. figure 23, p. 48).

As per Claim 10: Barrio discloses, *The method of claim 1, further comprising selectively switching between the emulation context* (See Case Switch program) *for running the emulator on the subject processor, a target execution context for executing target code produced by the emulator on the subject processor, and the native context where the subject code runs natively in the subject processor* (e.g.. p. 18, sec. 2.1, and p. 43, paragraph 'The getContext()…'))

As per Claim 11: Barrio discloses, *The method of claim 10, wherein both the native context and the emulation context employ a single image of the subject code* (i.e., testing an emulator. Note: when testing an emulator by running against a sample input code, versus a trusted emulator, the running code for two emulators must be the same, i.e., *employ a single image of the subject code*).

As per Claim 13: Barrio discloses, *The method of claim 1, comprising selecting between two or more verification modes, and dividing the subject code into the plurality of blocks according to the selected verification mode* (refer to "basic block' in the reference, e.g., sec. 2.2).

As per Claim 14: Barrio discloses, *The method of claim 13, comprising repeating the executing and comparing for each of the plurality of blocks* (refer to the program of p. 18).

As per Claim 15: Barrio discloses, *The method of claim 14, wherein in a first verification mode each block comprises a single instruction of subject code; in a second verification mode each*

block comprises a basic block comprising a sequence of instructions from a unique entry instruction to a unique exit instruction; and in a third verification mode each block comprises a group block comprising a plurality of the basic blocks (refer to program in p, 18, and see sections 2.1, 2.2, and 2.4).

As per Claim 16: Barrio discloses, *The method of claim 1, comprising: dividing a large segment of the subject code into a plurality of smaller blocks, each block containing one or more instructions from the large segment of subject code; and performing a verification comparison at a block boundary between each pair of consecutive neighbouring blocks in the plurality of blocks* (refer to program in p, 18, and see sections 2.1, 2.2, and 2.4).

As per Claim 17: Barrio discloses, *The method of claim 16, comprising: providing the subject processor in the emulation context, where control of the processor rests with the emulator, performing program code conversion on a current block BBn to produce a corresponding block of converted target code, and patching an immediately preceding block of subject code BBn-1 with a return jump; executing a context switch routine to enter the native context, and executing the immediately preceding block of subject code BBn-1 natively by the subject processor, such that the executing terminates with the return jump; executing a context switch routine to return to the emulation context, and performing the verification comparison by comparing a native machine state representing the subject processor following execution of the immediately preceding block BBn-1 with an emulated machine state representing a virtual model of the subject processor held by the emulator following execution of the immediately preceding block BBn-1; executing a context switch to a target execution context, and modelling execution of the target code corresponding to the current block of subject code BBn in the virtual model of the*

subject processor held by the emulator, thereby leaving the virtual model in a machine state representing the end of the current block BBn; and repeating the above for each subsequent block in the plurality of blocks, unless the verification comparison reveals an error in the program code conversion.

See p. 24-25, "Testing the emulator", where the emulator is a CPU emulator (p. 26) and including CPU emulator core (p. 42. 43, and 44)).

As per Claim 18: Barrio discloses, *The method of claim 17, further comprising restoring the immediately preceding block BBn-1 to remove the return jump* (only control flow of instruction operations, in a jump table).

As per Claim 19: Barrio discloses, *The method of claim 1, further comprising: selecting a block of the subject code; executing the block of subject code on the subject processor through the emulator; and appending a return jump to the block of subject code, and executing the block of subject code natively on the subject processor terminating with the return jump, such that the return jump returns control of the processor to the emulator* (e.g., see Fig. 13, p. 34).

As per Claim 100: Claim recites a system of verifying program code conversion performed by an emulator, where the performance in the claims has the patentable weight be equivalent to the limitations that are recited in the steps of claim 1. See rejection addressed in Claim 1 above.

As per Claim 101: Claim recite the system of verifying program code conversion performed by an emulator, where the further performance in the claim has the patentable weight be equivalent to the limitations that are recited in the steps of claim 4. See rejection addressed in Claim 4.

As per Claim 102: Claim recite the system of verifying program code conversion performed by an emulator, where the further performance in the claim has the patentable weight be equivalent to the limitations that are recited in the steps of claim 13. See rejection addressed in Claim 13.

As per Claim 103: Claim recite the system of verifying program code conversion performed by an emulator, where the further performance in the claim has the patentable weight be equivalent to the limitations that are recited in the steps of claim 15. See rejection addressed in Claim 15.

As per Claim 104: Claim recite the system of verifying program code conversion performed by an emulator, where the further performance in the claim has the patentable weight be equivalent to the limitations that are recited in the steps of claim 16. See rejection addressed in Claim 16.

As per Claim 105: Claim recite the system of verifying program code conversion performed by an emulator, where the further performance in the claim has the patentable weight be equivalent to the limitations that are recited in the steps of claim 19. See rejection addressed in Claim 19.

As per Claims 106-111: Claims recite a computer programming product of verifying program code conversion performed by an emulator, where the performance in the claims has the patentable weights be equivalent to the limitations that are recited in the steps of claims 1, 3-7, 10-11, 13-19. See rejection addressed in Claims 1, 3-7, 10-11, 13-19.

Conclusion

5. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ted T. Vo whose telephone number is (571) 272-3706. The

examiner can normally be reached on 8:00AM to 4:30PM. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Y. Zhen can be reached on (571) 272-3708.

The facsimile number for the organization where this application or proceeding is assigned is the Central Facsimile number **571-273-8300**.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

TTV
September 23, 2009

/Ted T. Vo/
Primary Examiner, Art Unit 2191